

# THE SERVERSIDE

Web Tech  
SET08101

Simon Wells

s.wells@napier.ac.uk

<http://www.simonwells.org>

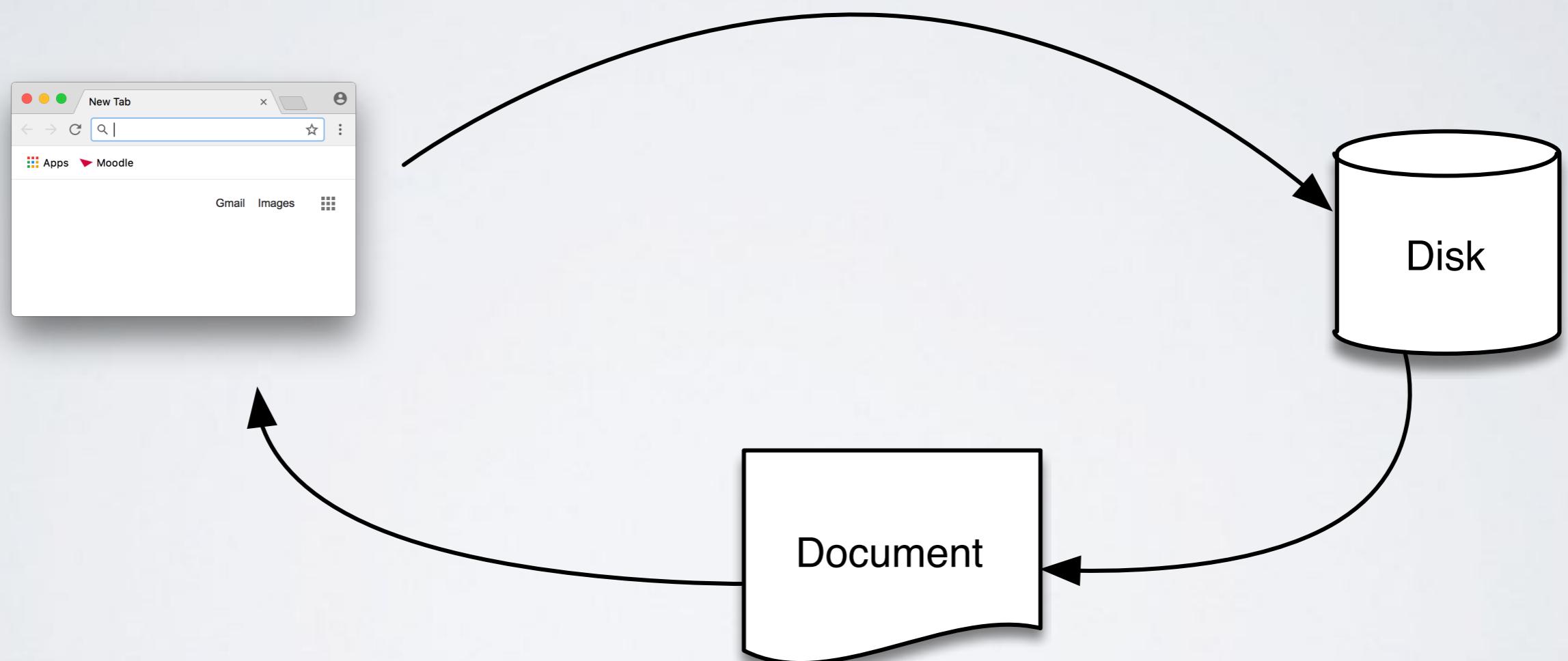
# TL/DR

- Developing **client** sites locally is all well & good, but it doesn't help us to share our work with others.
- Or to do more **dynamic & social** stuff
- To do that we need **servers**

# AIMS

- At the end of this (sub-section) of the topic you will be able to:
  - Know about servers (hardware & software)
  - Where the servers are
  - What they do
  - How to do stuff on the server side (instead of just in the client)

# LOCAL DEVELOPMENT



# DEPLOYMENT

- Need to store our web pages where they are **accessible** to our users
  - Accessible: client can connect to the storage location and request specific pages
  - NB. Other meanings of accessible
- The accessible location is known as a server
- Overloaded terminology: server can refer to **hardware & software**
- Server Hardware
- For the web there is software to listen for requests from web clients and provide an appropriate response



# SERVER HARDWARE

- Basically, any computer that can connect to the Internet
- From raspberry pi (not minimal) to huge clusters/supercomputers & everything inbetween
- Network Connection: Internet = (basically) TCP/IP



# SERVER SOFTWARE

- HTTP Server Software:

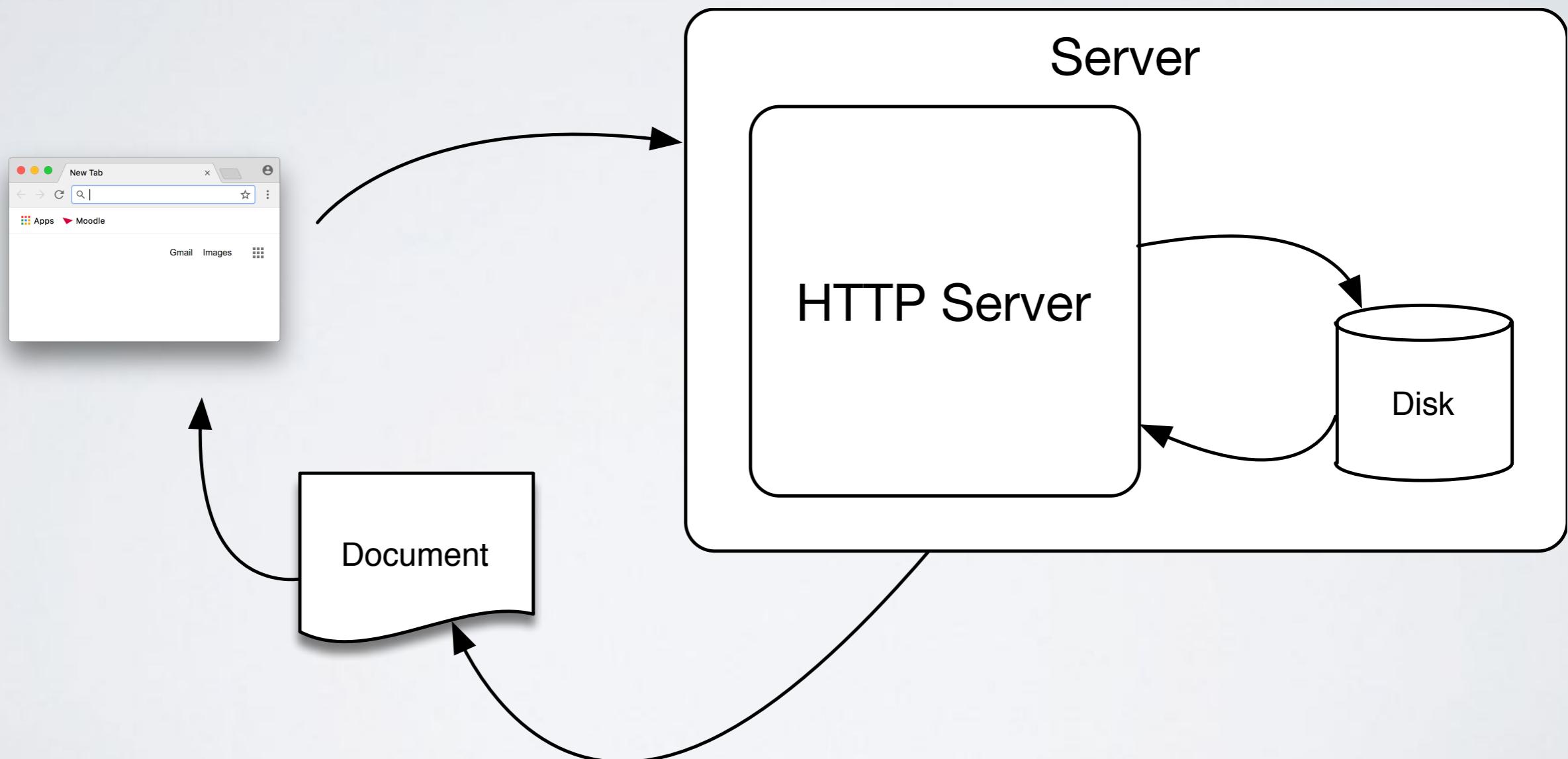
- nginx
- apache
- IIS
- Lighttpd (“lighty”)
- Others?



# CLIENTS & SERVERS

- An established architecture for organising communication between devices & software
- Client makes a **request**
- Server **listens** for requests then makes a **response**
  - Unless you're using the xwindow server in which case the relationship is back to front
- Server is a **nexus** - multiple clients can connect to it - server can relay data between clients

# CLIENT SERVER ARCHITECTURES



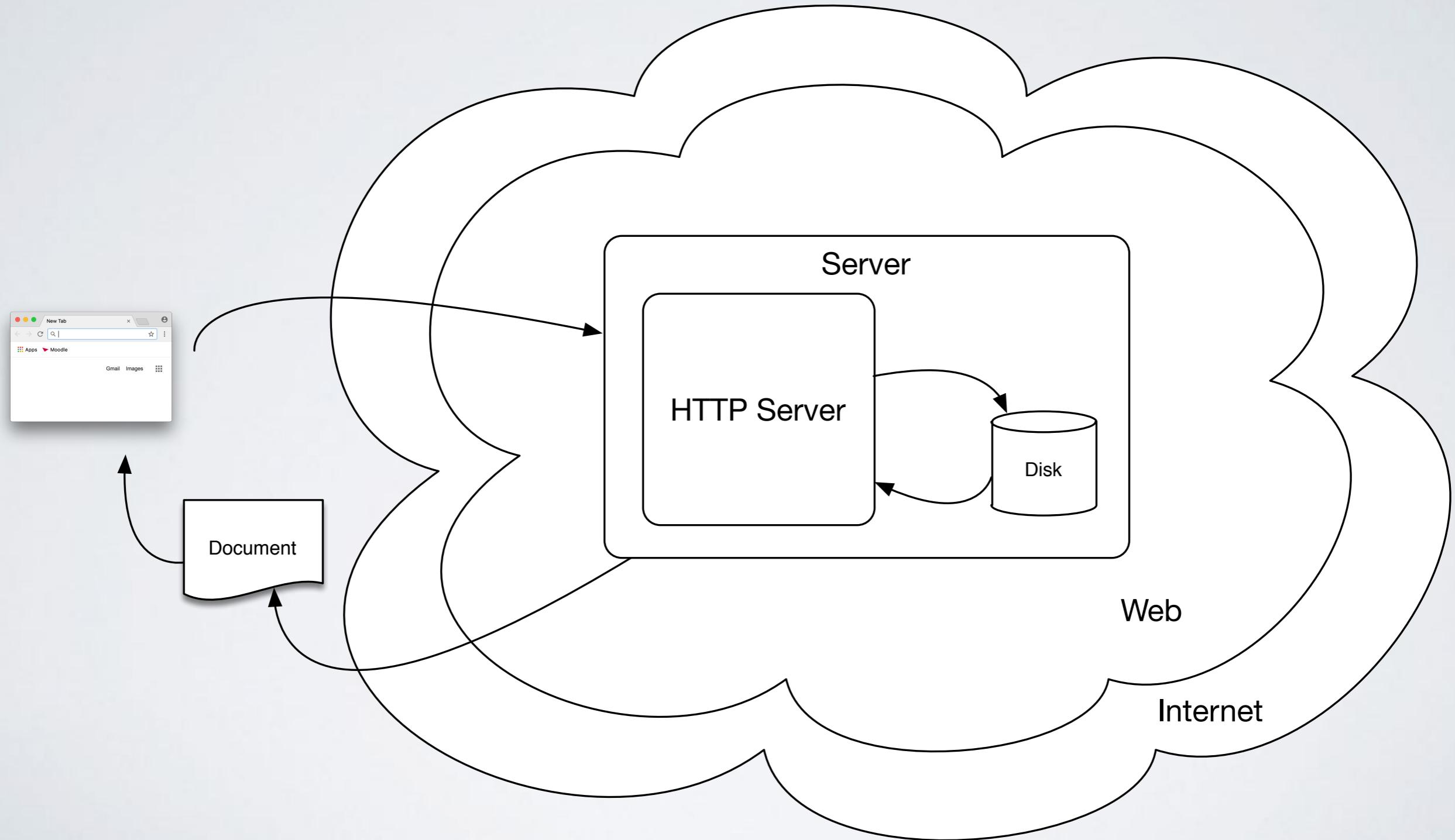
# WEB SERVING

- Server needs to be on the Internet: IP Address
- Server needs to be an HTTP server: Listens for connections using the HTTP protocol (RFC 2616)

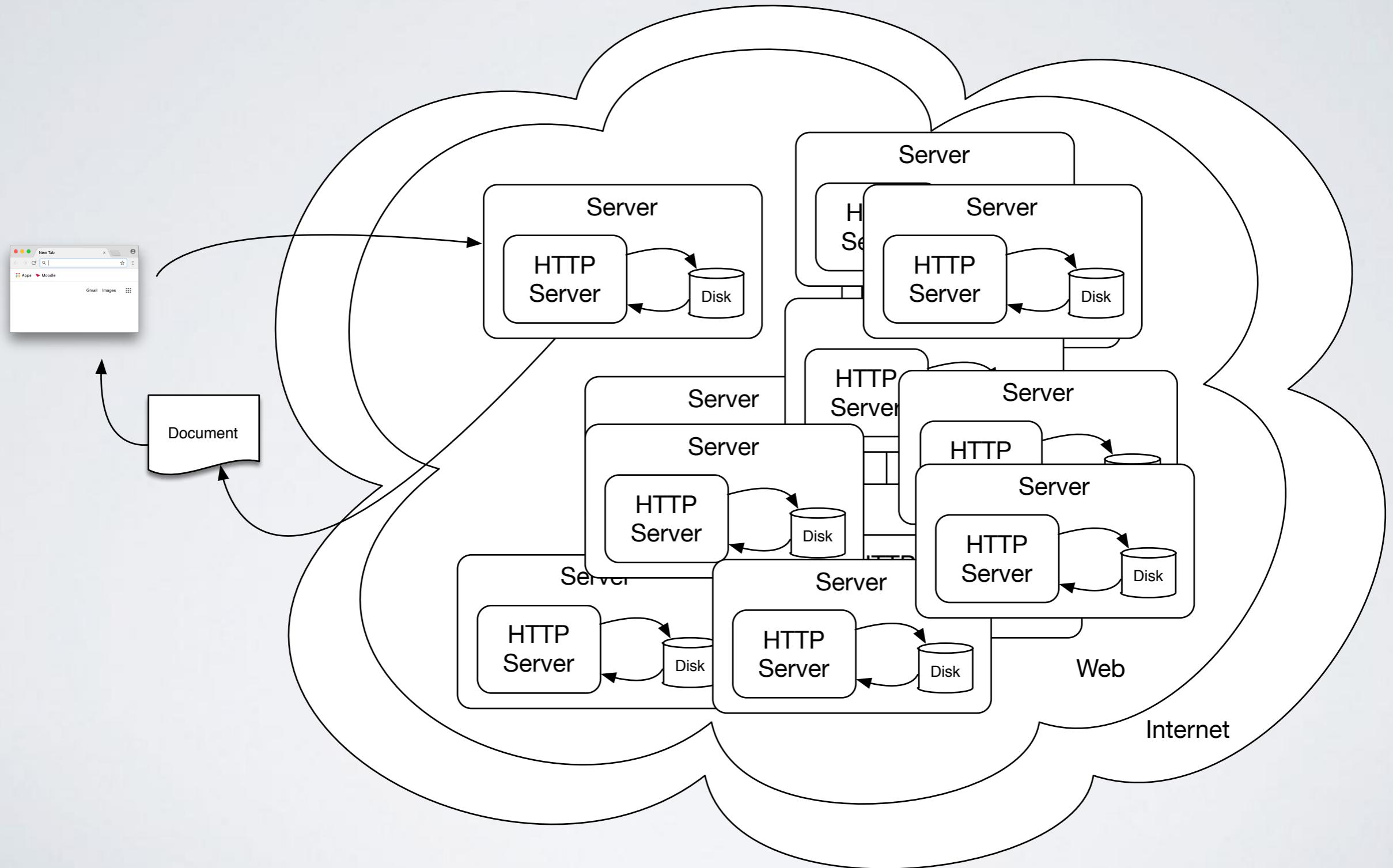
# HTTP PROTOCOL

- Defines how (primarily text) messages for transferring HyperText should be:
  - formatted & transmitted
  - responded to by servers & clients upon receipt of a given message
  - HTTP is a core standard of the Web (along with HTML)
  - Technically: a stateless, application layer protocol for communicating between distributed systems

# WEB ARCHITECTURE



# WEB SCALE

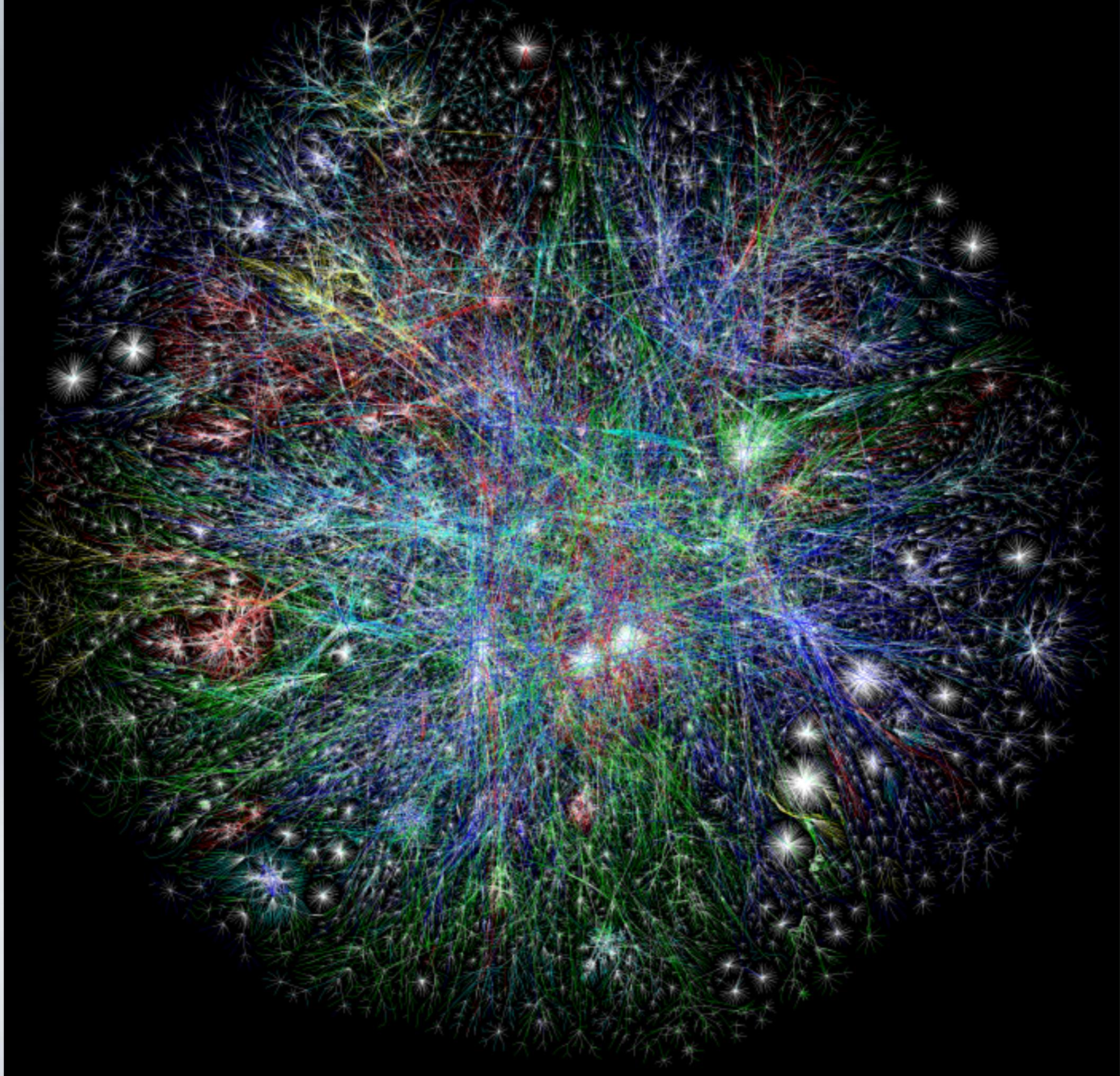


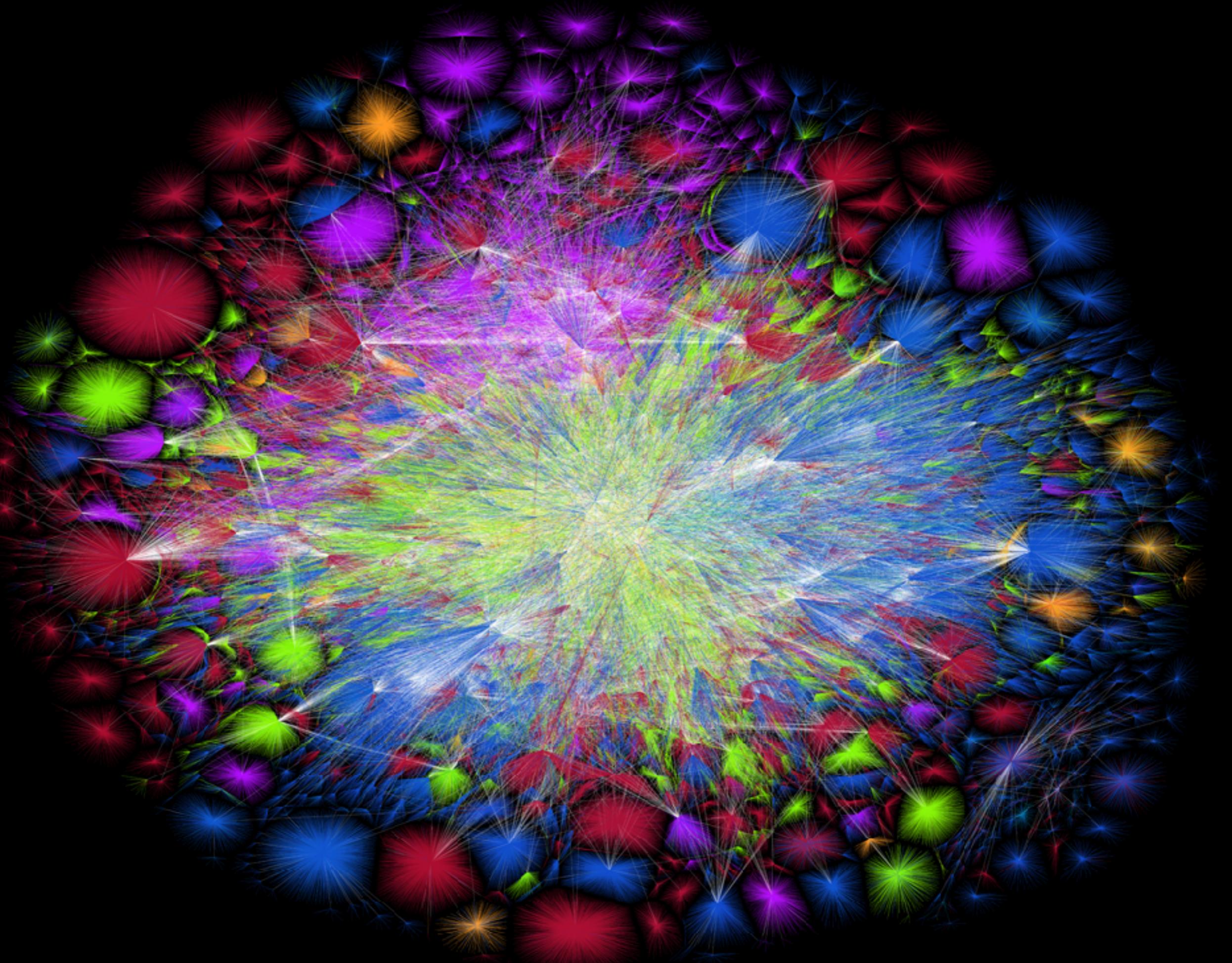
# THERE'S MORE...

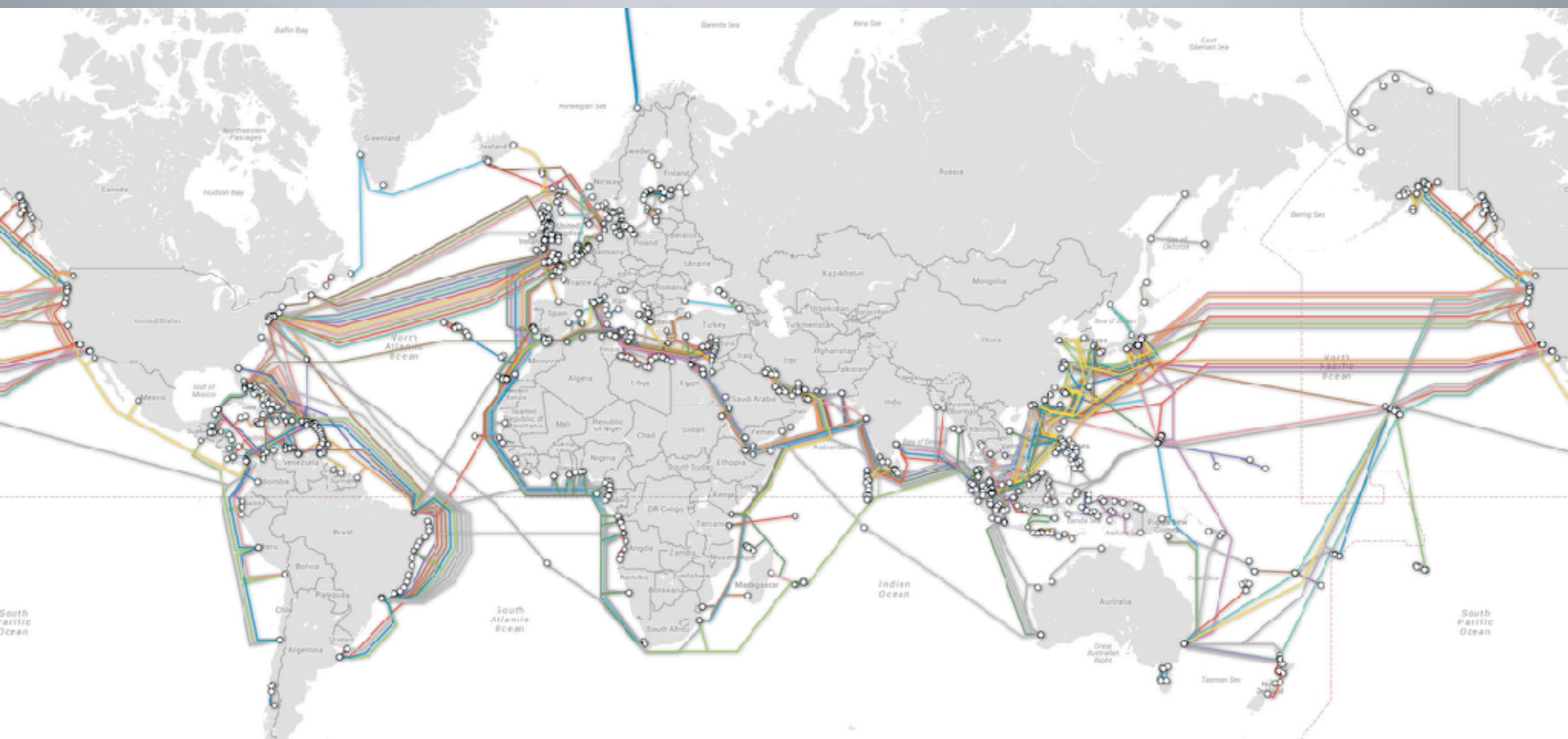
- Actually more complicated than just a bunch of web servers:
  - Each server may actually be part of a cluster of machines with incoming messages distributed between them (load balancing)
  - Datastores
  - Additional protocol servers: Web server might palm off certain jobs to dedicated machines, e.g. media streaming
  - Plus: Web caches, DNS, Time servers — anything else...?

# HOW BIG IS THE WEB?

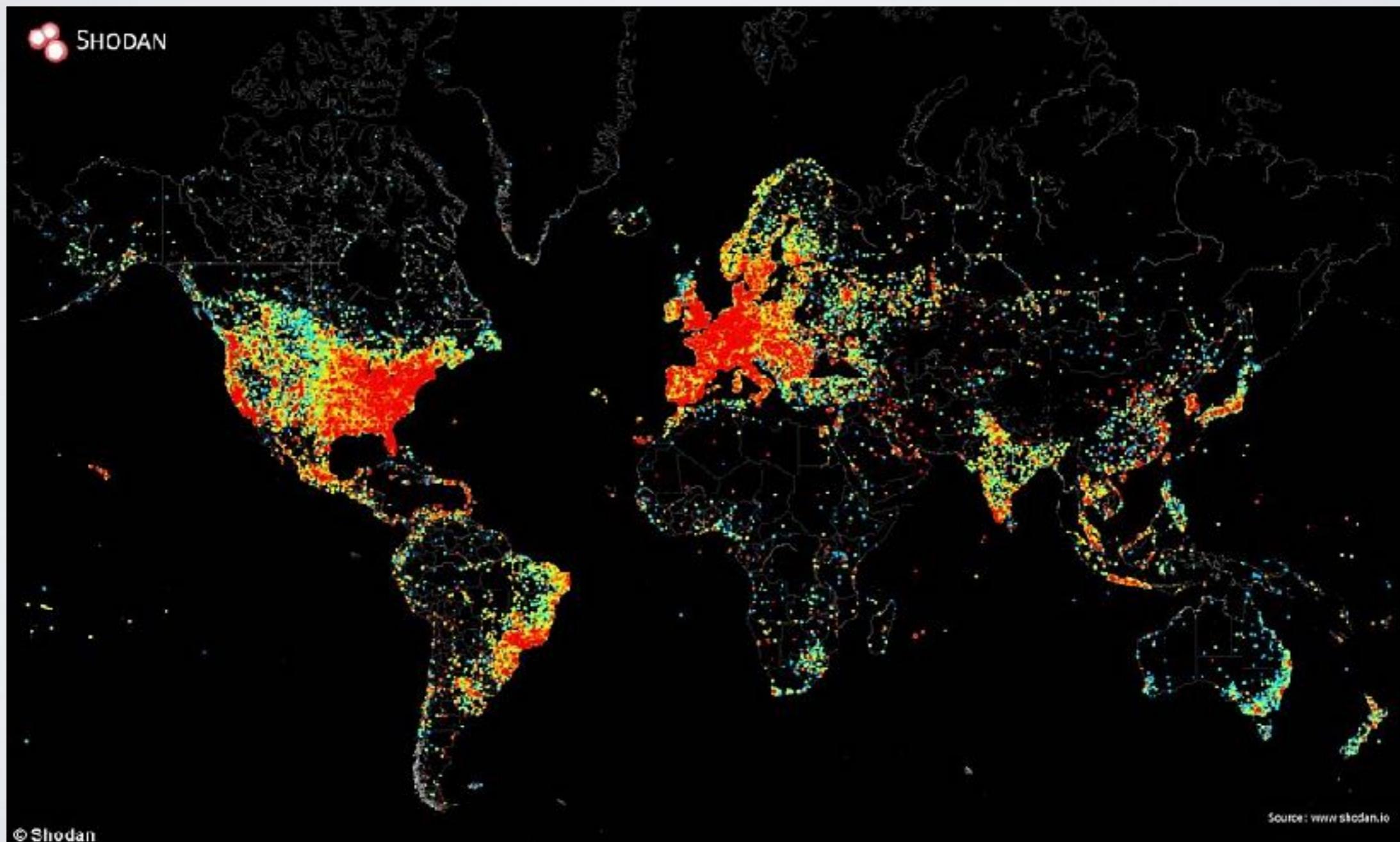
- Difficult to say for sure...
- Internet Live Stats [<http://www.internetlivestats.com/>]
- Cisco (visual networking initiative) estimates data transmission of 2 zettabytes per year across the Internet
  - 1 ZB = ~36,000 years of HD video
- Washington Post estimated (2015) 305B web pages
- Google estimated 30 Trillion unique pages







# INTERNET CONNECTED DEVICES









# WHERE DO THE SERVERS HIDE?

- Not all web servers are part of the public web
- Where else might we find servers?

# DEPLOYING OUR OWN SITES

- Neocities
  - Old School, free, static site hosting (in style of Geocities)
  - <https://neocities.org/>
- GitHub Pages
  - Static hosting direct from your GitHub repository
  - <https://pages.github.com/>

# SUMMARY

- We now know all about servers ;)



# NEXT

- More Node.JS & RESTful Design